Clustering Search Engine At Petra Christian University Library Using Suffix Tree Clustering

Andreas Handojo
Informatics Engineering Department
Faculty of Industrial Technology
Petra Christian University
Email: handojo@petra.ac.id

Adi Wibowo
Informatics Engineering Department
Faculty of Industrial Technology
Petra Christian University
Email: adiw@petra.ac.id

Jemmy Lay Santo
Informatics Engineering Department
Faculty of Industrial Technology
Petra Christian University
Email: m26405118@john.petra.ac.id

Abstract—**-At Petra Christian University Library, book searching engine system is using a common keyword matching as search queries. So, the user must enter the correct keyword and this case often takes a long time for users to find book that needed. Therefore, required an application that's capable display the search results that classified into a group/cluster. Which is each cluster will contain the documents that have the same classification base on keyword that been input by user in order to assist users in perform book searching. In this study, have been build an application to classify the document search results into a group of documents that have the same classification/cluster. Input to the cluster obtained from regular search results then the classification/clustering done by using Suffix Tree Clustering (STC) uses the document phrase.**

Based on the testing, the resulting cluster has been able to cluster documents that correlate or have the same classification with the keyword that input by user. The average time that needed to process for each document is 0.1139 seconds.

Keywords- **Suffix Tree Clustering, Suffix Tree, Clustering, Searching, Library**

I. INTRODUCTION

Information retrieval is concerned with representing, searching, and manipulating large collections of electronic text and other human-language data [4]. In general, the result from search systems on the web showing a long list of documents, then the user should look for from start to finish finding the desired document. This long list of document that build by search results sometimes make user confused and more difficult to find the desired document and also consume a lot of time.

The library of Petra Christian University, right now still use the same kind of method. So its makes user more difficult to find the book that he/she wants among thousands of books that available on the library. Besides of that user also must input the correct keyword to give a correct input to the searching engine.

Therefore it is necessary to build a clustering searching engine, where the general searching results will be accepted as an input and the engine will create a group of documents with the same classification (as a cluster), so this will be facilitate users in finding the desired document or book.

In performing the clustering, this application will used Suffix Tree Clustering (STC) that uses phrase to identify a set of documents that share a common phrase and use this information to create clusters and classifying its contents to the user using web application.

II. SUFFIX TREE

Suffix Tree (also called PAT tree or, in an earlier form, position tree [5]) is a data structures covering problems in strings (a series of character) to be able to analyzed quickly. If $txt=t_1t_2\ldots t_i\ldots t_n$ was a string, so $T_i=t_it_{i+1}\ldots t_n$ is suffix from txt, starting from position of i, example: [1]

T_1	=	mississippi = txt
T_2	=	ississippi
T_3	=	ssissippi
T_4	=	sissippi
T_5	=	issippi
T_6	=	ssippi
T_7	=	sippi
T_8	=	ippi
T_9	=	ppi
T_{10}	=	pi
T_{11}	=	i
T_{12}	=	(empty)

Suffix tree could used to find a substring, pat [1 ... m] in O (m). There are n(n +1)/2 substrings in txt [1 ... n] and be more surprisingly that the suffix tree could formed in time O (n). The addition of single characters in the txt increase the number of substrings with n + 1. In the example above, if suffix is zero (empty) the process will be sorted as follows:

T11	=	i
T8	=	ippi
T5	=	issippi
T2	=	ississippi
T1	=	mississippi
T10	=	pi
T9	=	ppi
T7	=	sippi
T4	=	sissippi
T6	=	ssippi
T3	=	ssissippi

It is clear that some of them give the same prefix. There are substrings starting with 'i', 'm', 'p', and 's', but all

that starts with 'is' in fact start with 'issi'. Two or more of the same prefix gives the same path from the root of the suffix tree.

```
Tree - - > |- - -mississippi
           |- - -i- -> |- - -ssi- -> |- - -ssippi
           |           |             |- - -ppi
           |           |- - -ppi
           |- - -s- ->|- - -si- ->  |- - -ssippi
           |           |             |- - -ppi
           |           |- - -i- - >  |- - -ssippi
           |                         |- - -ppi
           |- - -p- ->|- - -pi
                      |- - -i
```

III. CLUSTERING

Clustering method is a method that has the ability to analyze and classify documents automatically. Clustering technique usually use words and documents as a collection of words without considering any order or so-called bag of words.

Cluster analysis is an effort to find a group of objects that represent a character of the same or nearly the same (similar) between one object with another object in a group and have a differences (not similar) with the objects in other groups. Of course the similarities and differences were obtained based on information provided by these objects and their relationships (relationship) between them.

IV. SUFFIX TREE CLUSTERING [3]

Suffix Tree Clustering (STC) is the first algorithm that uses phrase so the process is much simpler than other algorithms. STC is an incremental algorithm, the calculations is a linear time complexity O(n) and meet the criteria for document clustering. [2]

Suffix Tree Clustering has three step of logic:

A. Document Cleaning

The sentence is marked and not remove, the original string documents is remaining in the keep.

B. Basic Cluster Identification

The identification of the basic cluster can be expressed as the making of an inverted index of the phrases from the document collections. This can be done efficiently with a data structure called a suffix tree. Suffix tree of a string 'S' is a compact tree that composed from the phrase that have suffix 'S'. Which is the document will be considered as a string of the words, not as a character. Therefore, the suffix will be consists of the one or a whole word.

Examples suffix tree of a set of strings 'cat ate cheese', 'mouse ate cheese too', and 'cat ate mouse too'. Node of the suffix tree is depicted in a circle (Figure 1). Each suffix-node has one or more boxes that are connected to indicate a string of origin. The first number in each box indicates the string origin, while the second number suffix indicating which of the label string, the suffix-node it.

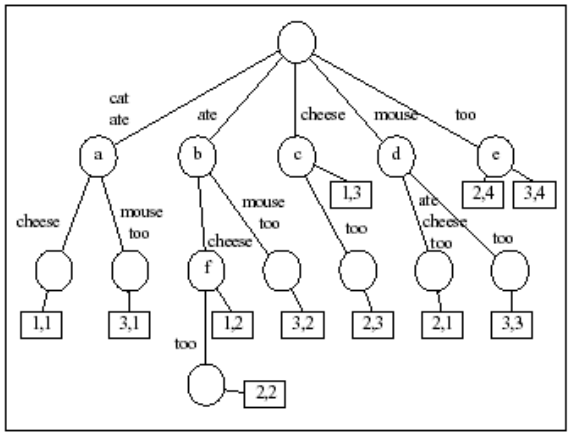

Figure 1. Suffix tree from string 'cat ate cheese', 'mouse ate cheese too', and 'cat ate mouse too'

Each node of the suffix tree represents a group of the documents and a phrase that is common to all. The label of the node represents a common phrase, a collection of the tagging documents its suffix node is a descendant of the node to form a group of documents (Table 1).

Therefore, basic cluster (consisting of the 2 or more documents) appear as a node in a suffix tree.

TABLE I. SIX NODES AT FIGURE 1 AND THE BASIC CLUSTER CORRESPONDENT

Node	*Phrase*	*Documents*
A	*cat ate*	1,3
B	*ate*	1,2,3
C	*cheese*	1,2
D	*mouse*	2,3
E	*too*	2,3
F	*ate cheese*	1,2

Each basic cluster is marked by a score that is a function of the the number of documents that are owned and words that make the phrase. Score from basic cluster B with the phrase P can be written:

$$s(B) = |B|. f(|P|) \quad (1)$$

Where | B | is the number of documents in basic cluster B
s is the score than basic cluster B,
|B| is the number of documents in basic clusters B,
|P| is the number of words in P that has a score is not zero.

Words that include stop list or appears too few (3 or less) or too many (more from 40% from a collection of documents) were given score 0. The function f penalizes the 1 word phrase, linear to long-word phrases by 2 to 6 and became constant for longer phrases.

C. *Basic Cluster Combination*

Documents may share more than one phrase. As a result, a collection of documents from different basic clusters may overlap and may even be the same. a binary similarity measurement between the basic clusters is defined by the overlap from collection of documents. Given two basic clusters B_m and B_n, with size $|B_m|$and $|B_n|$ each, and $|B_m \cap B_n|$ represents the number of documents to the two basic clusters, defined similarity from B_m and B_n to be 1 if:

$$|B_m \cap B_n|/|B_m| > 0{,}5 \text{ and} \quad (2)$$
$$|B_m \cap B_n|/|B_n| > 0{,}5$$

In contrast, the similarity is defined by 0.

V. IMPLEMENTATION AND TESTING

There are several main steps in from this system, which is:

- Documents Cleaning.
- Basic Cluster identification.
- Combination Basic Cluster Association.

Figure 3, will display the flowchart from the application

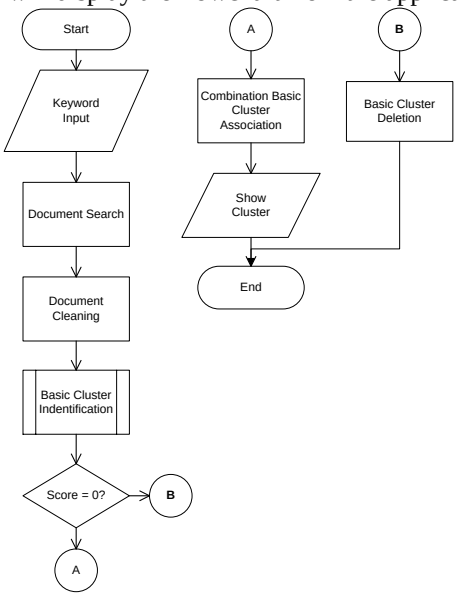

Figure 2. General System *Flowchart* from the Application

A. *Document Cleaning*

Document Cleaning is the first step to get the basic cluster from the documents that find from the common search results. In the process of cleaning document, punctuation in the title is removed. After completing the cleaning process then the result will be send to the basic cluster identification stage.

B. *Basic Cluster Identification*

The second step is to identify the basic cluster. At this stage there are a series of basic cluster identification process will be carried out to obtain basic cluster. This process will include sorting process, tree making process and the process of calculating basic cluster score. From Nodes that formed, the number of documents owned by each node will be calculated, if there are nodes that have a number of documents as much as 2 or more then the node is stored as a cluster basic. Then, each basic cluster obtained score will be calculated

In the process of calculating a score, administrator could determine the lower and upper limit for the words that included in stop list or words that have a zero score. Every word in the sentence (title) will be taken and application will calculated the number of documents that containing the word. If the number is greater than the lower limit or smaller than the upper limit or included in the stop word the word is given a zero score. Then the words in basic cluster that doesn't have zero score will be counted and also the number of documents containing the basic cluster will be count. After getting the number of documents (|B|) and the words in basic cluster that has zero score then score than score from each basic cluster can be calculated (2.2)

C. *Basic Cluster Association*

After getting the basic cluster, the next step is to combine these basic clusters. At this step, application will look for any relationship from existing basic cluster. Any basic clusters that have a relationship or similarity, the association value will be set by 1, if not then the value will be set by 0. To analyze the correlation from two basic cluster, the number of documents from each cluster will be count as $|B_m|$ and $|B_n|$, and $|B_m \cap B_n|$ will be indicates as the number of documents that owned together by that two basic clusters. Two basic clusters are said that have similarities or relationships if:

$$|B_m \cap B_n|/|B_m| > 0{,}5 \text{ and}$$
$$|B_m \cap B_n|/|B_n| > 0{,}5$$

Besides that, the value will be set as 0.

After getting the basic cluster relationship or similarity, then if the basic cluster included in stop list or has zero score, the basic cluster is removed. The examination will be given input query using keyword 'microcontroller'. The result of cluster base obtained as shown at tables 2.

TABLE II. RESULT OF BASIC CLUSTER WITH QUERY ' MICROCONTROLLER'.

No.	*Base Cluster*
1	68hc12 microcontroller
2	68hc12/11 microcontrollers
3	and the microcontroller
4	design of embedded systems using 68hc12/11 microcontrollers

5	digital
6	digital signal processing and the microcontroller
7	embedded systems using 68hc12/11 microcontrollers,
8	microcontroller
9	microcontrollers
10	of embedded systems using 68hc12/11 microcontrollers
11	processing and the microcontroller
12	signal processing and the microcontroller
13	systems using 68hc12/11 microcontrollers
14	the microcontroller
15	using 68hc12/11 microcontrollers

After getting the basic clusters, and based on the number of documents and words that form the phrase base clusters, the computed score than base cluster B with the phrase P using equation 1.

The score that calculate from table 2 base cluster number 11 'processing and the microcontroller' is as follows:

- Number of words = 4.
- The number of words that have a zero score = 3 ('and', 't he', and 'microcontroller'). The word 'and' and 'the' has zero score because included in stop word while the word 'microcontroller' has zero score because it has a to much appearance number (more than 40% from the number of documents with the query 'microcontroller'). So the number of words that do not have score zero is (|P|) = 1.
- The number of documents (|B|) = 2.

Thus, the obtained score for the base cluster 'processing and the microcontroller' as shown in table 2 Number 11 is 0. Furthermore, than base clusters obtained by the selected bas e clusters that has non-zero score to serve as a cluster. The result of cluster with query ‘microcontroller’ can be shown on table 3.

TABLE III.
RESULT OF CLUSTER WITH QUERY 'MICROCONTROLLER'.

No.	*Cluster*
1	68hc12/11 microcontrollers
2	design of embedded systems using 68hc12/11 microcontrollers
3	digital signal processing and the microcontroller
4	embedded systems using 68hc12/11 microcontrollers
5	signal processing and the microcontroller
6	systems using 68hc12/11 microcontrollers
7	using 68hc12/11 microcontrollers

The application from the clustering search engine using suffix tree can be shown on Figure 3.

The examination result from system performance that calculate the relation between number of documents that have been process and process time required it’s presented by graphic (as shown on Figure 4).

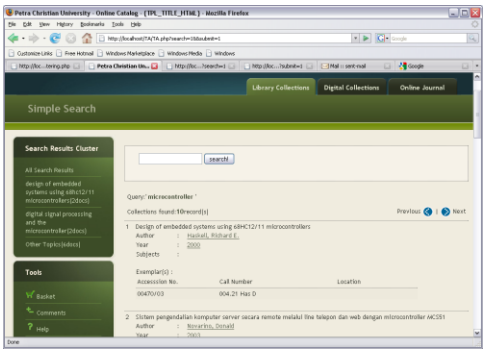

Figure 3. Library Clustering Search Engine

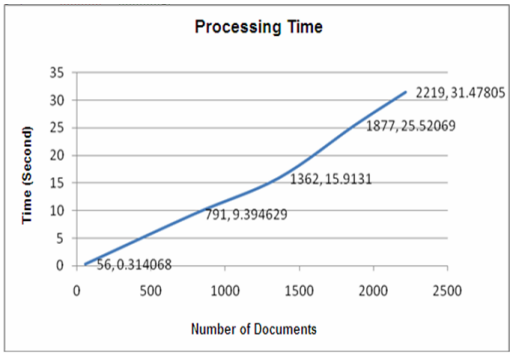

Figure 4. Correlation between Time and Number of Documents

VI. CONCLUSION

Based on the testing result on the system, we can conclude several things as follows:

- The cluster that generated by the system is good enough, in a sense to classify the documents that correlate to each other or have the same classification.
- The processing time is increase relatively equal to the number of documents.
- The process to combination the basic cluster is requires the longest time between any other processes.

REFERENCES

[1] Allison, Lloyd. (n.d.). *Suffix trees*. Retrieved January 21, 2011, from http://allisons.org/ll/AlgDS/Tree/Suffix/

[2] Zamir, Oren, Etzioni Oren, Omid Madani, Richard M Karp. (1997). *Fast and Intuitive clustering of web Documents.* KDD-97 Proceedings.

[3] Zamir, Oren, and Etzioni Oren. (1998). *Web document clustering: a feasibility demonstration.* Washington: Department of Computer Science and Engineering University of Washington Seattle.

[4] Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack. *Information Retrieval: Implementing and Evaluating Search Engines.* MIT Press, Cambridge, Mass., 2010.

[5] *Suffix Tree*. Retrieved January 21, 2011. *http://en.wikipedia.org/wiki/Suffix_tree*